Chapter_01

- 1.1 Professional software development
- 1.2 Software engineering ethics
- 1.3 Case studies
- 1.1 Professional software development
    - 1.1.1 Software Engineering:
    - 1.1.2 Software Engineering Diversity:
    - 1.1.3 Internet Software Engineering:
- 1.2 Software Engineering Ethics: Key Points:
- 1.3 Case studies:

Chapter_02

- 2.1 Software Process Models
- 2.2 Process Activities
- 2.3 Coping With Change
- 2.4 Process Improvement
- 2.1 Software Process Models
    - 2.1.1 Waterfall
    - 2.1.2 Incremental
    - 2.1.3 Integration and Configuration (Reuse)
- 2.2 Process Activities (SDVE)
    - 2.2.1 Specification (Requirements Engineering)
    - 2.2.2 Design and Implementation
    - 2.2.3 Validation (System Testing)
    - 2.2.4 Evolution
- 2.3 Coping With Change
    - 2.3.1 Prototyping
    - 2.3.2 Incremental Delivery
- 2.4 Process Improvement

Chapter_03

- 3.1 Agile methods
- 3.2 Agile Development techniques
- 3.3 Agile Project Management

- 3.4 Scaling Agile Methods
- 3.1 Agile methods
- 3.2 Agile Development techniques
    - 3.2.1 User Stories
    - 3.2.2 Refactoring
    - 3.2.3 Test-first Development (TDD)
    - 3.2.4 Pair Programming
- 3.3 Agile Project Management
- 3.4 Scaling Agile Methods
    - 3.4.1 Practical problems with agile methods
    - 3.4.2 Agile and plan-driven methods
    - 3.4.3 Agile methods for large systems
    - 3.4.4 Agile methods across organizations

Chapter_04

- 4.1 Functional and non-functional requirements
- 4.2 Requirements engineering processes
- 4.3 Requirements Elicitation
- 4.4 Requirements specification
- 4.5 Requirements validation
- 4.6 Requirements change
- 4.1 Functional and non-functional requirements
    - 4.1.1 Functional requirements
    - 4.1.2 Non-functional requirements
- 4.2 Requirements engineering processes
- 4.3 Requirements Elicitation
    - 4.3.1 Requirements Elicitation Techniques
    - 4.3.2 Stories and Scenarios
- 4.4 Requirements specification
    - 4.4.1 Natural language specification
    - 4.4.2 Structured specifications
    - 4.4.3 Use cases
    - 4.4.4 The software requirements document
- 4.5 Requirements validation
- 4.6 Requirements change
    - 4.6.1 Requirements management planning
    - 4.6.1 Requirements change management

Chapter_05

- 5.1 Context models
- 5.2 Interactions model
- 5.3 Structural models
- 5.4 Behavioral models
- 5.5 Model-driven architecture
- 5.1 Context models
- 5.2 Interactions model
  - 5.2.1 use case modelling
  - 5.2.1 Sequence Diagrams
- 5.3 Structural models
  - 5.3.1 Class Diagrams
  - 5.3.2 generalization
  - 5.3.3 Aggregation
- 5.4 Behavioral models
  - 5.4.1 Data-driven modelling
  - 5.4.2 Event-driven modelling
  - 5.4.3 model-driven Engineering (MDE)
- 5.5 Model-driven architecture (MDA)

Chapter_06

- 6.1 Architecture Design Decisions
- 6.2 Architectural Views
- 6.3 Architectural patterns
- 6.4 Application Architectural
- 6.1 Architecture Design Decisions
- 6.2 Architectural Views
- 6.3 Architectural patterns
  - 6.3.1 Layered Architecture
  - 6.3.2 Repository Architecture
  - 6.3.3 Client-Server Architecture
  - 6.3.4 Pipe and Filter Architecture
- 6.4 Application Architectural
  - 6.4.1 Transactional Processing Systems
  - 6.4.2 Information Systems
  - 6.4.3 Language Processing Systems

Chapter_07

- 7.1 Object oriented design using UML
- 7.2 Design patterns
- 7.3 Implementation issues
- 7.4 Open source development
- 7.1 Object oriented design using UML
  - 7.1.1 System context and interactions
  - 7.1.2 Architectural design
  - 7.1.3 Object class identification
  - 7.1.4 Design models
  - 7.1.5 Interface specification
- 7.2 Design patterns
- 7.3 Implementation issues
  - 7.3.1 Reuse
  - 7.3.2 Configuration management
  - 7.3.3 Host-target development
- 7.4 Open source development
  - 7.4.1 Open source licensing

Chapter_08

- 8.1 Development testing
- 8.2 Test-driven development
- 8.3 Release testing
- 8.4 User testing
- 8.1 Development testing:
  - 8.1.1 Unit testing
  - 8.1.2 Choosing unit test cases
  - 8.1.3 Component testing
  - 8.1.4 System testing
- 8.2 Test-driven development
- 8.3 Release testing
  - 8.3.1 Requirements-based testing
  - 8.3.2 Scenario testing
  - 8.3.3 Performance testing
- 8.4 User testing

Chapter_09

- 9.1 Evolution processes
- 9.2 Legacy systems
- 9.3 Software maintenance
- 9.1 Evolution processes:
- 9.2 Legacy systems:
    - 9.2.1 Legacy system management:
- 9.3 Software maintenance:
    - 9.3.1 Maintenance prediction:
    - 9.3.2 Software reengineering:
    - 9.3.3 Refactoring:

Chapter_10

- 10.1 Dependability properties
- 10.2 Socio-technical systems
- 10.3 Redundancy and diversity
- 10.4 Dependable processes
- 10.5 Formal methods and dependability
- 10.1 Dependability properties
- 10.2 Socio-technical systems:
    - 10.2.1 Regulation and Compliance
- 10.3 Redundancy and Diversity
- 10.4 Dependable Processes
- 10.5 Formal Methods and Dependability

Chapter_11

- 11.1 Availability and reliability
- 11.2 Reliability requirements
- 11.3 Fault-tolerant architectures
- 11.4 Programming for reliability
- 11.5 Reliability measurement
- 11.1 Availability and Reliability
- 11.2 Reliability Requirements
    - 11.2.1 Reliability Metrics
    - 11.2.2 Nonfunctional Reliability Requirements

    - 11.2.3 Functional Reliability Specification
- 11.3 Fault-tolerant Architectures
    - 11.3.1 Protection Systems
    - 11.3.2 Self-monitoring Architectures
    - 11.3.3 N-version Programming
    - 11.3.4 Software Diversity
- 11.4 Programming for Reliability
- 11.5 Reliability Measurement
    - 11.5.1 Operational Profiles

Chapter_12

- 12.1 Safety-critical systems
- 12.2 Safety requirements
- 12.3 Safety engineering processes
- 12.4 Safety cases
- 12.1 Safety-Critical Systems
- 12.2 Safety Requirements
    - 12.2.1 Hazard Identification
    - 12.2.2 Hazard Assessment
    - 12.2.3 Hazard Analysis
    - 12.2.4 Risk Reduction
- 12.3 Safety Engineering Processes
    - 12.3.1 Safety Assurance Processes
    - 12.3.2 Formal Verification
    - 12.3.3 Model Checking
    - 12.3.4 Static Program Analysis
- 12.4 Safety Cases
    - 12.4.1 Structured Arguments
    - 12.4.2 Software Safety Arguments

Chapter_15

- 15.1 The reuse landscape
- 15.2 Application frameworks
- 15.3 Software product lines
- 15.4 Application system reuse
- 15.1 The Reuse Landscape
- 15.2 Application Frameworks
- 15.3 Software Product Lines

- 15.4 Application System Reuse
    - 15.4.1 Configurable Application Systems
    - 15.4.2 Integrated Application Systems

Chapter_18

- 18.1 Service-oriented architecture
- 18.2 RESTful services
- 18.3 Service engineering
- 18.4 Service composition
- 18.1 Service-Oriented Architecture (SOA)
- 18.2 RESTful Services
- 18.3 Service Engineering
- 18.4 Service Composition

Chapter_19

- 19.1 Socio-technical systems
- 19.2 Conceptual design
- 19.3 System procurement
- 19.4 System development
- 19.5 System operation and evolution
- 19.1 Socio-technical Systems
    - 19.1.1 Emergent Properties
    - 19.1.2 Nondeterminism
    - 19.1.3 Success Criteria
- 19.2 Conceptual Design
- 19.3 System Procurement
- 19.4 System Development
- 19.5 System Operation and Evolution
    - 19.5.1 System Evolution